FALSIFYBENCH: Evaluating Inductive Reasoning in LLMs with Rule Discovery Games

Anonymous ACL submission

Abstract

Large language models (LLMs) are increasingly deployed as autonomous agents in scientific tasks. Yet whether these systems can effectively engage in forms of inductive reasoning relevant to scientific discovery remains an open question. In this work, we introduce FALSIFYBENCH, an evaluation framework for hypothesis-driven reasoning inspired by the classic Wason 2-4-6 task, in which agents must discover hidden semantic properties by iteratively proposing examples and receiving feedback. This task captures key elements of scientific reasoning: hypothesis generation, evidence gathering, and belief revision in response to both confirming and disconfirming evidence. Our evaluation of 12 LLMs across model families and scales shows that reasoning models are generally stronger scientific reasoners than instruction-tuned models, although no model comes close to optimal performance. The primary driver of success is the capacity for negative testing: models that actively seek to falsify their hypotheses consistently outperform those that primarily seek confirmation. Moreover, a fine-grained turn-level analysis, neglected in previous work, reveals that failure is tied to identifiable patterns in how models navigate the hypothesis space.

1 Introduction

Large language models (LLMs) are becoming increasingly capable across domains associated with scientific work, including literature review (Mialon et al., 2024; Wei et al., 2025), software engineering and programming (Jimenez et al., 2024; Jain et al., 2025), mathematics (Mirzadeh et al., 2025; Glazer et al., 2025), and expert-level knowledge (Wang et al., 2024; Rein et al., 2024; Center for AI Safety et al., 2026). These advances have motivated a growing body of work on autonomous agents capable of performing scientific research in an end-to-end manner (Lu et al., 2024; Gottweis et al., 2025). As these systems become increasingly integrated into scientific workflows, a fundamental question arises: can they carry out the key forms of inductive reasoning required for scientific research? Central among these is the ability to generate, evaluate, and revise hypotheses in light of new evidence. A critical component of this process is *falsification*, whereby a hypothesis is rejected in favor of an alternative when it is contradicted by the evidence.

One of the best-known and most widely used paradigms for studying this process in humans is the Wason 2-4-6 task (Wason, 1960). Participants are shown a triple of numbers (e.g., "2, 4, 6") and asked to discover the hidden rule underlying it, denoted here as R. They do so by proposing new triples and receiving feedback from the experimenter on whether each triple fits the rule. Once sufficiently confident that they have discovered R, participants state their hypothesis (denoted here as H). When the rule to be discovered is more general than the one suggested by the initial example (as in Wason's original task, where R is "Any increasing sequence of three numbers"), success rates are low. This is because participants tend to form overly specific hypotheses (e.g., "Even numbers increasing by two"). In such cases, generating triples con-

Relation	Falsification of H	Strategy
$H = R$	None	None
$H \subset R$	$x \notin H \wedge x \in R$	$-$
$H \supset R$	$x \in H \wedge x \notin R$	$+$
Partial overlap	$(x \in H \wedge x \notin R) \vee (x \notin H \wedge x \in R)$	$+/-$
Disjoint	$x \in H \vee x \in R$	$+/-$

Table 1: Set relations between a possible hypothesis H and the true rule R, where x denotes a test triple. Each relation has specific *falsification* cases that force the rejection of H in favor of a new hypothesis. The highlighted case ($H \subset R$) corresponds to the Wason 2-4-6 task and the games in FALSIFYBENCH: here, only triples outside H (negative tests, $-$) can reveal that R is broader than H, whereas for other relations triples inside H (positive tests, $+$) may falsify H.

sistent with specific hypotheses (e.g., "8, 10, 12") cannot reveal that those hypotheses are too narrow and falsify them. This behavior has been termed *Confirmation bias* and has traditionally been interpreted as an irrational tendency that runs counter to scientific reasoning.

Later work by Klayman and Ha (1987) has made a more nuanced distinction between *positive testing*, in which one seeks evidence expected to confirm the current hypothesis, and *negative testing*, in which one seeks evidence expected to disconfirm it. Crucially, they showed that positive testing can also lead to the falsification of a hypothesis and that the effectiveness of these two strategies depends on the relation between H and R. Specifically, they showed that a positive test strategy can falsify a hypothesis H when it is broader than the hidden rule R ($H \supset R$), while it fails to falsify H whenever this is a strict subset of R ($H \subset R$). In the latter case, *negative testing* is the only strategy that can conclusively falsify H. Table 1 provides a summary of all possible relations between H and R, together with the type of test that can conclusively falsify H in each case.[1]

In this work, we introduce a generalized version of the Wason task in which LLMs must discover hidden semantic properties rather than numerical rules. In this semantic version, we exploit taxonomic relations to build games emphasizing the $H \subset R$ configuration, where negative testing is the optimal strategy and which corresponds to Wason's original case. This configuration is particularly interesting because it mirrors a recurring pattern in the history of science, where accepted theories later turn out to be special cases of broader ones (e.g., Newtonian mechanics as a limiting case of relativity). This parallel motivates our focus on the falsification and revision of overly narrow hypotheses through negative testing as a key component of scientific reasoning, and one worth testing in systems such as LLMs. At the same time, following Klayman and Ha (1987), we do not assume that negative testing is universally superior. Instead, we examine how LLMs handle hypothesis testing across different relations between H and R and apply the most appropriate normative analysis to each case, as described below.

Our main contributions are as follows: (1) we introduce FALSIFYBENCH[2], a benchmark for hypothesis-driven reasoning grounded in semantic taxonomies, which operationalizes a key component of scientific reasoning: the ability to actively generate, and potentially falsify, a current hypothesis in order to control for it; (2) we evaluate 12 LLMs across multiple model families and scales, showing that reasoning models generally behave as stronger scientific thinkers than instruction-tuned models in our framework, although no model approaches optimal performance; and (3) we run a fine-grained turn-level analysis, made possible by our evaluation setting, showing that failures reflect identifiable patterns in how models navigate the hypothesis space.

2 Related Work

Inductive reasoning is central to a wide range of human activities, including empirical scientific research, and has therefore become a key focus in evaluations of reasoning in LLMs.

Early studies have tested this ability in sequence-to-sequence models using synthetic benchmarks. Chollet (2019) introduced the Abstraction and Reasoning Corpus (ARC), which tests models on grid transformation tasks, in which they must infer abstract transformations from a few example input-output pairs and learn to apply the correct rule to unseen grids. Lake and Baroni (2018) take a similar approach, testing the capability to learn mappings from commands to action strings, and testing their application to novel, unseen compositions of sequences.

Recent work has increasingly focused on understanding how LLMs perform inductive reasoning beyond task performance. Jin et al. (2026) showed that chain-of-thought reasoning (Wei et al., 2022) does not universally improve induction and can even reduce performance. Li et al. (2025) introduced a framework for evaluating and explaining inductive reasoning in LLMs by disentangling induction and deduction steps. Louapre (2026) examined how LLMs behave in an inductive game, explicitly mapping models' strategies to human-like traits that influence performance, and connecting their setting to scientific discovery.

Within this line of research, our work is closely related to studies that also draw inspiration from Wason's 2-4-6 task. Banatt et al. (2024) introduce WILT, a multi-turn inductive logic benchmark where hidden rules are boolean functions involving

[1] We include a detailed discussion of cases where H can be conclusively falsified, with visual examples, in Appendix A.

[2] The code and datasets used in this work will be made publicly available, upon acceptance, under a CC BY 4.0 license.

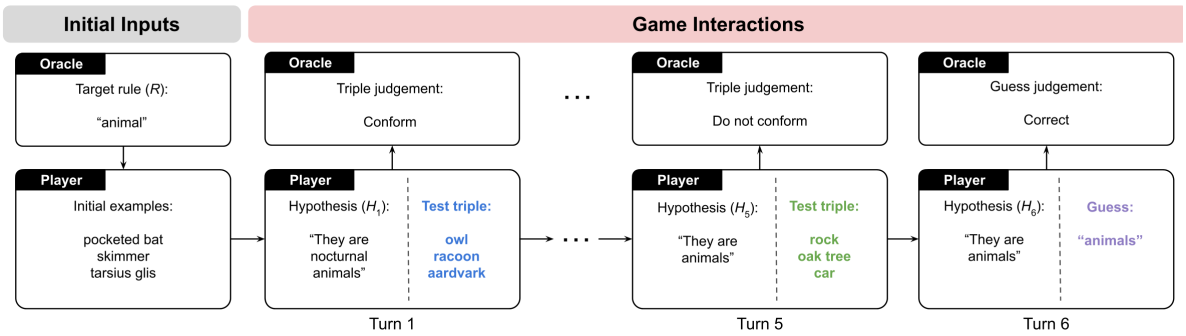


Figure 1: FALSIFYBENCH game structure. The player is given three items (e.g., pocketed bat, skimmer, and tarsius glis) and must identify their shared hidden target rule (e.g., "they are animals") by proposing additional test triples and receiving feedback from the oracle ("Conform" vs. "Do not conform"). Before proposing a test triple, the player makes explicit the hypothesis about the hidden rule being tested. In **positive tests**, the proposed triples are expected to satisfy this hypothesis, whereas in **negative tests**, they are expected to falsify it. The process continues until the player is sufficiently confident that they have identified the hidden rule and articulates it as a **guess**. If the stated hypothesis is correct, the game ends; otherwise, the player proceeds with a new test triple.

three variables; Jhaveri et al. (2026) use rules about number properties and explicitly study Confirmation bias in LLMs, applying interventions studied in humans to LLMs to encourage falsificatory behavior. Neither study explicitly takes into account the normative framework introduced by Klayman and Ha (1987): they neither sample triples from a rule that is a strict subset of the general target rule nor track how the hypotheses formed by models at each turn relate to the target rule. In our experiment, we instead address this limitation by sampling triples of terms from nodes that are strict taxonomic descendants of R, thereby making negative testing the better strategy for identifying it.

3 Method

3.1 Experiment Overview

In our experiment, each LLM served in two roles: as the oracle, which answered queries, and as the player, which attempted to discover the hidden rule. In direct analogy with Wason's original experiment, the oracle provided the player with an initial triple of terms sampled from a subordinate semantic category (e.g., "vertebrate"), which was itself included in a more general target category (e.g., "animal"). The more general target category served as the target rule R. By iteratively proposing new examples and receiving feedback, the player accumulated evidence, testing its current hypothesis either positively or negatively, until it could identify R. An example of a real game is shown in Figure 1. More specifically, on each turn, the player proposed three items along with a hypothesis and a rationale for why those items were chosen. The oracle responded with the string "Conform" or "Do not conform" depending on whether the set of items satisfied the target rule R. When sufficiently confident, the player submitted a final answer, and the oracle responded with either "Correct" or "Incorrect". Thus, the player was prompted to use `Test` actions to gather information and refine its hypotheses, and to use `Guess` actions once enough information had been collected. When the player submitted an incorrect guess, it learned that its current hypothesis was wrong and was prompted to perform new actions to gather additional evidence until it was confident enough to make a new guess. A game ended when either the player submitted a correct guess or the maximum turn limit was reached (we set the maximum number of turns to 20).

We chose semantic categories, rather than numerical sequences, as the domain of our experiment for two reasons: LLMs are very likely to have encountered the classic 2-4-6 task during pretraining, and semantic taxonomies allow us to define a rich set of candidate concepts with increasingly specific subconcepts. The categories were extracted from WordNet (Miller, 1995), which enables the construction of a large number of such games by selecting general concepts and sampling from their descendants. From this large space, we curated a set of 100 representative games spanning different taxonomic depths and semantic domains.

Each game was defined by two semantic categories: the first was the general category that served as the target hidden rule R, while the second was

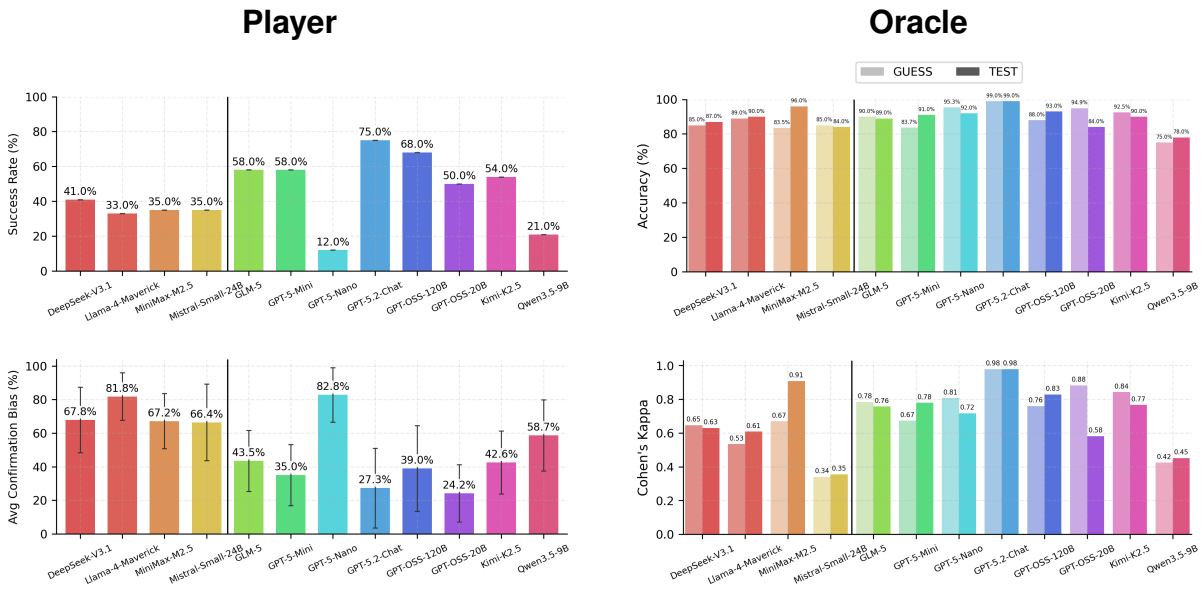


Figure 2: Experimental results for 12 LLMs on FALSIFYBENCH. Models to the left of the vertical black line are **instruction-tuned** variants, while models to the right are **reasoning** variants. **Top-left:** Success rate represents the percentage of correctly identified target rules. **Bottom-left:** Confirmation bias measures the proportion of queries that confirm rather than falsify the current hypothesis (lower is better). **Top-right:** Oracle accuracy against human annotations on `Test` and `Guess` turns. **Bottom-right:** Cohen's κ agreement between oracle and human annotations on `Test` and `Guess` turns.

a sampling category, S, a descendant of R from which the initial examples were drawn. For instance, if R = "animal" and S = "vertebrate", a possible set of items was E = {"pocketed bat", "skimmer", "tarsius glis"}. The player observed these items and formed an initial hypothesis, H_1. Because the items were drawn from the node S, which was deep in the taxonomy, semantic categories close to that region tended to appear more plausible than very general ones, like R, so players typically began with a hypothesis that was more specific than the true rule.[3] When H was more specific than R, *positive testing* always returned positive feedback. Making progress therefore required *negative testing*: only by proposing items outside the current hypothesis could the player determine that R was broader than hypothesized and revise the hypothesis upward toward it.

Overall, we selected 7 high-level WordNet categories as candidate target rules: "animal", "artifact", "body part", "food", "location", "plant", and "worker". For each, we identified all descendant subcategories and sampled triples from their leaves, yielding 17,905 total games. Since many WordNet subcategories involve low-frequency vocabulary, we manually filtered the subcategory pool and created a curated set of 100 games spanning 5 categories: "animal", "artifact", "body part", "food", and "plant". Full details on game construction and the complete list of selected sampling categories appear in Appendix B. The results we report in this paper are collected on this curated set of 100 games.

The two roles assumed by each LLM (i.e, player and oracle) were instantiated through separate API calls. This allowed us to evaluate models not only on hypothesis-driven reasoning, but also on their ability to provide semantic judgments consistent with the taxonomic knowledge encoded in WordNet. We forced both agents to produce actions in JSON format to constrain the space of their possible outputs. We provide more details on the exact prompts and JSON schema used in our experiments in Appendix C.

3.2 Metrics

We evaluate performance of LLMs on FALSIFYBENCH using the following metrics.

Player Success Rate. We computed the percentage of games in which the player correctly identified the target rule R. This served as the primary

[3]This pattern mirrors Wason's findings, where participants exposed to the sequence "2-4-6" favored overly specific rules such as "increasing even numbers" instead of the true rule "any increasing sequence".

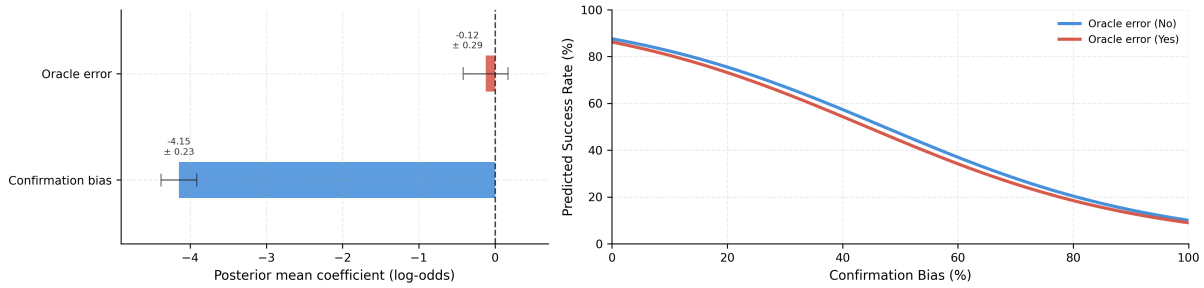


Figure 3: Results of the Bayesian mixed-effects logistic regression (n = 1,200 games). **Left:** Posterior means (± 2 posterior SD) for the two fixed-effect predictors. Confirmation bias is a strong negative predictor of success, while oracle error rate has a negligible effect. **Right:** Predicted probability of success as a function of Confirmation bias, shown separately for games in which the oracle made at least one error and games in which it did not. The near-identical curves indicate that oracle error rate is not the primary driver of player outcomes.

measure of task performance, indicating whether the model could converge on the correct hypothesis.

Player Confirmation Bias. We tracked the proportion of positive tests among all tests performed within a game to assess whether models instantiated a general positive testing strategy, as humans often do in the original Wason task. We then averaged this proportion across games and refer to it as Confirmation bias. By construction, in our experiment, positive tests were expected to be largely ineffective at falsifying the current hypothesis, making this metric a direct measure of the extent to which a model pursues an uninformative strategy.

Oracle Taxonomic Knowledge. Since each LLM served in both roles (i.e., as player and as oracle), we quantified the oracle's taxonomic knowledge to assess whether potentially inaccurate feedback could contribute to poor player performance. To this end, for each game and each model, we randomly sampled one triple from the `Test` turns and one hypothesis from the `Guess` turns.[4] A human annotator then labeled the oracle's feedback for each triple and each guess according to whether it conformed to R. The annotator was given access to the NLTK library (Bird and Loper, 2004) and to search engines to ensure that annotations accurately reflected the taxonomic relations encoded in WordNet. We evaluated oracle performance by measuring its accuracy and agreement with the human annotations, using Cohen's κ.

[4]With 100 games and 12 models, this yields 1200 `Test` instances and 1200 `Guess` instances.

4 Performance on FALSIFYBENCH

Our evaluation across 12 models revealed substantial variation in player and oracle performance, which we analyze below.

Player Performance and Confirmation Bias. Reasoning models generally achieved higher success rates than their instruction-tuned counterparts. The best instruction-tuned model, DeepSeek-V3.1, reached only 41% success rate, while top reasoning models achieved 75% (GPT-5.2-Chat) and 68% (GPT-OSS-120B), with the remaining instruction-tuned models clustering between 33% and 35%. Notably, success rates remained far from saturated even for the best model, leaving substantial room for improvement. The advantage of reasoning models did not uniformly extend to the smallest ones: GPT-5-Nano (12%) and Qwen3.5-9B (21%) performed worse than all instruction-tuned models, suggesting that extended thinking alone was insufficient at smaller scales. Additionally, we found a strong correlation between success rates and Confirmation bias: models that predominantly used positive testing consistently underperformed those that actively tested outside their current hypothesis (Spearman $\rho = -0.779$, $p = 0.003$). This pattern held across both model families and scales, with stronger reasoning models maintaining substantially lower Confirmation bias ratios (GPT-5-Mini: 39.0%, GPT-5.2-Chat: 27.3%, GPT-OSS-20B: 24.2%) compared to instruction-tuned models, which clustered between 66% and 82%.

Oracle Quality. In terms of oracle performance with respect to human annotation, raw accuracy was moderately high across all models, with most LLMs clustering between 85% and 95%. Cohen's

κ revealed more meaningful variation, spanning from 0.34 (Mistral-Small-24B, `Guess` turns) to 0.98 (GPT-5.2-Chat, both conditions), indicating that some models provided less reliable taxonomic judgments than others. However, models with comparable oracle performance showed substantially different success rates: for instance, MiniMax-M2.5, GLM-5, and GPT-5-Nano achieved broadly similar oracle κ scores (0.67–0.91, 0.78–0.76, and 0.81–0.72, respectively) yet succeeded on 35%, 58%, and 12% of games respectively. This dissociation suggested that oracle quality was not sufficient for player success. We analyzed this further below.

Disentangling Player Reasoning from Oracle Quality. We aimed to better assess whether player strategy or oracle quality were determinant for the player's success. To this end, we fit a Bayesian mixed-effects logistic regression at the game level ($n = 1{,}200$), with Confirmation bias and oracle error rate as fixed-effect predictors and LLM identity as a random effect.[5] Confirmation bias was used in the model as a continuous predictor in $[0, 1]$, representing the proportion of positive tests within a game. Oracle error rate was modeled as a binary variable, taking the value 1 if the oracle's response disagreed with the human annotator on either the sampled `Test` or `Guess` turn for that game, and 0 otherwise.[6] As shown in Figure 3 (left), Confirmation bias was a strong negative predictor of success (posterior mean $= -4.15$, OR $= 0.016$, posterior SD $= 0.116$), while oracle error rate showed no credible independent effect (posterior mean $= -0.12$, OR $= 0.885$, posterior SD $= 0.147$), with a credible interval spanning zero. Moreover, Figure 3 (right) showed near-identical predicted success curves across oracle error conditions, suggesting that failures were driven primarily by the model's own Confirmation bias rather than by the quality of the oracle's feedback. The residual between-model variance captured by the random effect ($\hat{\sigma}_u = 0.553$) indicates that LLM identity provides additional explanatory information beyond the two predictors. This reflects the factors that differentiate the overall quality of LLMs.

[5] See Appendix D.2 for complete model details and results.

[6] Although an exhaustive human annotation of all turns across 1,200 games would enable us to calculate an exact continuous oracle error rate, we restricted our analysis to this binary estimate to keep the annotation effort manageable.

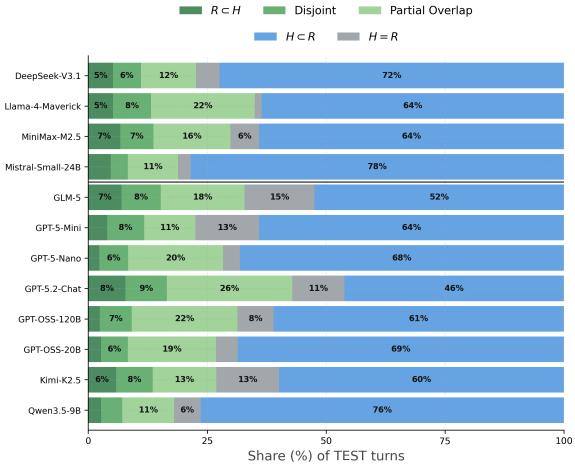


Figure 4: Distribution of relations between the current hypothesis H and target R across all `Test` turns, per model. The dominant configuration, $H \subset R$ (blue), is the only case in which positive testing is uninformative and negative testing is the only type of action that can result in a falsification of H. The green relations are ones where positive testing may also lead to a falsification of H. In the gray relation, $H = R$, no falsification is possible.

5 In-depth Turn Level Analysis

The Confirmation bias reported in Section 4 measures the *intent* behind a model's testing strategy, implicitly assuming that positive testing is a less optimal behavior than negative testing. In order to determine the effectiveness of a given strategy in rejecting a wrong hypothesis, it is necessary to understand the relationship between each current hypothesis H and the target rule R in games (see Table 1). Indeed, it is possible that the player would form hypotheses for which negative testing is suboptimal and positive testing should instead be favored. For this reason, we employed one of the models tested on FALSIFYBENCH as an offline LLM annotator, which annotates all `Test` turns of each game across all models with the set relation between H and R. To balance speed and annotation cost, we selected GPT-5-Mini as annotator. The exact prompt used to perform this annotation can be seen in Appendix C.

Hypothesis-Target Relations. Figure 4 shows the distribution of relations between H and R across all `Test` turns from all games for each model. The dominant configuration across all models is $H \subset R$ (Hypothesis in Target), consistent with the assumption we made when building FALSIFYBENCH: since the initial items are drawn from a

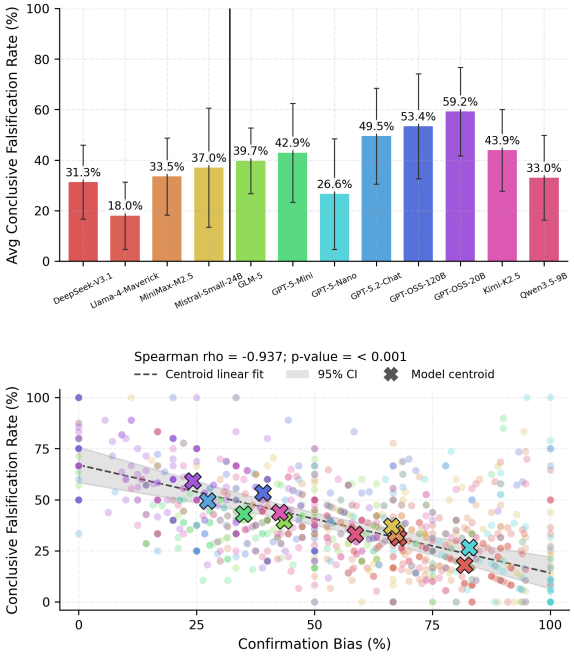


Figure 5: **Top:** Average conclusive falsification rate per model: the proportion of `Test` turns that yield a conclusive falsification given the relation between H and R at that turn. Models to the left of the vertical black line are **instruction-tuned** variants; models to the right are **reasoning** variants. **Bottom:** Scatter plot of Confirmation bias against conclusive falsification rate across all 12 models. The strong negative correlation (Spearman $\rho = -0.937$, $p < 0.001$) indicates that models that test more positively tend to reach fewer conclusive falsifications.

semantic category S which is a descendant of R, models will form hypotheses that are more specific than the true target and revise them upward, yielding a sequence of $H \subset R$ turns. For almost all models, the relation $H \subset R$ was present in more than 50% of all turns, with the exception of GPT-5.2-Chat, which exhibited a lower share of $H \subset R$ turns and a higher share of partial-overlap relations. Importantly, the $H \subset R$ configuration is the only one in which negative testing is the uniquely optimal strategy, and it remained the most common throughout games. In the cases where H is partially overlapping with or disjoint from R, both positive and negative testing may lead to a falsification of H (see Table 1).

Confirmation bias vs. Conclusive Falsification Rate. Having access to the full annotation of the relation between H and R in each `Test` turn, we could check for each turn whether a positive or negative test would yield a falsification of H. Thus, we measured the average proportion of tests yielding a falsification of H across games, called *conclusive falsification rate* (Figure 5, top). Reasoning models again outperformed instruction-tuned models on this measure, with GPT-OSS-20B (59.2%), GPT-OSS-120B (53.4%), and GPT-5.2-Chat (49.5%) achieving the highest rates. The lowest rates were again produced by Llama-4-Maverick (18.0%) and GPT-5-Nano (26.6%). As explained earlier, negative testing is not always guaranteed to be the better strategy for falsifying a wrong hypothesis; however, we show here that it was indeed mostly the case in FALSIFYBENCH: Figure 5 (bottom) shows the Confirmation bias against the conclusive falsification rate across all models, revealing a strong negative correlation (Spearman $\rho = -0.937$, $p < 0.001$). The strength of this correlation supports the idea that testing negatively (i.e., having lower Confirmation bias) was a reliable proxy for reaching a conclusive falsification of wrong hypothesis in our framework.

6 Failure Beyond Confirmation Bias

As shown in the previous sections, Confirmation bias is associated with failure in FALSIFYBENCH. Beyond this bias, which is related to the testing strategy players adopted, failure may also be attributable to the type of hypotheses they generated.

Failed games showed a systematically greater proportion of turns where the hypothesis was either disjoint from or partially overlapping with the target relative to successful games, indicating that models that failed did not follow a clear taxonomic path toward R.[7] Figure 6 (left) quantifies this difference by reporting, for each model, the mean gap in the proportion of partially overlapping and disjoint turns between failed and successful games. Positive values indicate that a given configuration is more prevalent in failed games. Asterisks denote statistically significant differences in the median number of turns of each type (Mann-Whitney U test, $p < 0.05$). Nearly all models exhibited a significantly higher proportion of partially overlapping turns in failed games, with reasoning models showing particularly pronounced differences (GPT-5.2-Chat: +27.0 pp, GPT-OSS-120B: +22.3 pp, GLM-5: +21.4 pp). This suggests that in failed games models frequently formed hypotheses that were similar to R but failed to converge on the correct rule. Disjoint turns were also significantly

[7] We show a full per-model decomposition of relation distributions across successful and failed games in Appendix E.2.

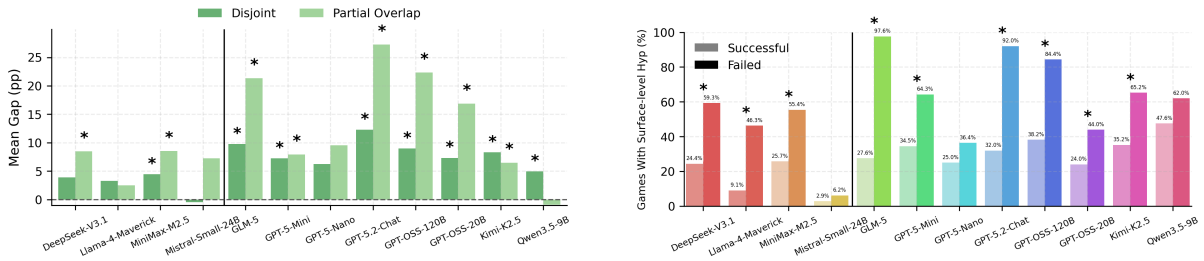


Figure 6: **Left:** Mean gap (in percentage points) in the proportion of Disjoint and Partial Overlap `Test` turns between failed and successful games, per model. Positive values indicate a configuration is more prevalent in failed games. Asterisks denote statistically significant differences in the median number of turns of each type between the two groups (Mann-Whitney U test, $p < 0.05$). **Right:** Percentage of successful and failed games in which at least one hypothesis generated across all turns was classified as focusing on surface-level linguistic features. Asterisks indicate models for which the proportion was significantly higher in failed than in successful games according to Fisher's exact test. In both plots, models to the left of the vertical black line are **instruction-tuned** variants; models to the right are **reasoning** variants.

more frequent in failed games for most models, though the effect was smaller in magnitude, indicating that while fully off-track hypotheses did occur, they were less prevalent than partially overlapping ones.

While these results provide evidence that, in failed games, models often did not fully grasp the taxonomic structure of the hypothesis space, we also aimed to qualitatively characterize the kinds of hypotheses they generated when diverging from the correct path. A notable case involved a focus on surface-level features of the words used in the game (e.g. "all items start with the letter a" or "all items have exactly three vowels in their names") rather than on semantic properties (such as, "is a fish" or "is a mammal"). To detect such cases, we built a heuristic classifier based on regular expressions.[8] Figure 6 (right) shows the percentage of games, both successful and failed, in which at least one hypothesis, across all game turns, was classified as focusing on surface-level features of items. Models for which a Fisher's exact test revealed a significantly higher proportion in failed games ($p < 0.05$) are denoted by an asterisk. This indicates that the tendency to form surface-level linguistic hypotheses was systematically associated with failure. Interestingly, for models with higher overall success rates, such as GLM-5, GPT-5.2-Chat and GPT-OSS-120B, nearly all failed games were characterized by the presence of at least one linguistic hypothesis (97.6%, 92.0% and 84.4% respectively). In contrast, weaker models such as GPT-5-Nano and Qwen3.5-9B showed no significant difference between successful and failed games (36.4% vs. 25.0% and 62.0% vs. 47.6% respectively), suggesting that these models generally struggled with the task and did not exhibit this specific failure mode.

[8] Additional details regarding the regular expression patterns used to build the classifier are reported in Appendix D.3.

7 Conclusion

We introduced FALSIFYBENCH, a benchmark for evaluating hypothesis-driven reasoning in LLMs through a semantic generalization of the Wason 2-4-6 task. Our evaluation of 12 models showed that reasoning models are generally stronger scientific thinkers than instruction-tuned models. However, no model came close to optimal performance, highlighting substantial room for improvement in the inductive reasoning capabilities of current LLMs. The primary driver of success was found to be the capacity for negative testing: models that actively seek to falsify their hypotheses consistently outperform those that seek to confirm them, and our turn-level analysis showed that disconfirmatory behavior is a reliable proxy for progress toward the correct hypothesis.

Finally, we showed that failure is also tied to how models explored the hypothesis space. In failed games, models tended to drift away from the target, forming hypotheses that were either disjoint from or only partially overlapping with the target rule, and were significantly more likely to propose rules based on surface-level properties of words rather than semantic categories.

8 Limitations

Our analysis uses WordNet as a proxy for semantic taxonomic knowledge. Although WordNet is a

well-established lexical resource created by expert lexicographers, it has several limitations. For example, its hierarchy reflects specific lexicographic decisions, it is English-centric, and it may contain basic semantic inconsistencies. We sought to overcome these limitations by focusing on concrete and fairly unambiguous superordinate categories.

Additionally, we collected human annotations for only one `Test` turn and one `Guess` turn per game. While this sampling strategy kept annotation effort manageable across 1,200 games, it forced us to operationalize oracle error as a binary variable. We acknowledge that this is a coarse approximation. Nevertheless, the high raw accuracy observed across models suggests that the probability of a randomly selected turn resulting in an incorrect oracle judgment is low. This means that our binary estimate is likely to have overestimated rather than underestimated the true oracle error rate, making our regression result conservative. Despite this, the effect of oracle error on player success remains negligible.

Finally, our analysis was inspired by the history of scientific discoveries, in which previously accepted hypotheses have been replaced by more comprehensive ones. We acknowledge, however, that this is only one aspect of the scientific process, and leave further exploration of this topic to future research.

References

Eryk Banatt, Jonathan Cheng, Skanda Vaidyanath, and Tiffany Hwu. 2024. Wilt: A multi-turn, memorization-robust inductive logic benchmark for llms. *Preprint*, arXiv:2410.10998.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Center for AI Safety, Scale AI, and HLE Contributors Consortium. 2026. A benchmark of expert-level academic questions to assess AI capabilities. *Nature*, 649:1139–1146.

François Chollet. 2019. On the measure of intelligence. *CoRR*, abs/1911.01547.

Elliot Glazer, Ege Erdil, Tamay Besiroglu, Diego Chicharro, Evan Chen, Alex Gunning, Caroline Falkman Olsson, Jean-Stanislas Denain, Anson Ho, Emily de Oliveira Santos, Olli Järviniemi, Matthew Barnett, Robert Sandler, Matej Vrzala, Jaime Sevilla, Qiuyu Ren, Elizabeth Pratt, Lionel Levine, Grant Barkley, and 5 others. 2025. Frontiermath: A benchmark for evaluating advanced mathematical reasoning in ai. *Preprint*, arXiv:2411.04872.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, Khaled Saab, Dan Popovici, Jacob Blum, Fan Zhang, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Pushmeet Kohli, and 15 others. 2025. Towards an ai co-scientist. *Preprint*, arXiv:2502.18864.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2025. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*.

Ayush Rajesh Jhaveri, Anthony GX-Chen, Ilia Sucholutsky, and Eunsol Choi. 2026. Failing to falsify: Evaluating and mitigating confirmation bias in language models. *Preprint*, arXiv:2604.02485.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. 2024. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*.

Haibo Jin, Peiyan Zhang, Man Luo, and Haohan Wang. 2026. Evaluating the inductive abilities of large language models: Why chain-of-thought reasoning sometimes hurts more than helps. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Joshua Klayman and Young Whee Ha. 1987. Confirmation, disconfirmation, and informa-tion in hypothesis testing.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2873–2882. PMLR.

Jiachun Li, Pengfei Cao, Zhuoran Jin, Yubo Chen, Kang Liu, and Jun Zhao. 2025. MIRAGE: Evaluating and explaining inductive reasoning process in language models. In *The Thirteenth International Conference on Learning Representations*.

David Louapre. 2026. Can llms play the game of science?

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. The ai scientist: Towards fully automated open-ended scientific discovery. *Preprint*, arXiv:2408.06292.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2024. GAIA: a benchmark for general AI assistants. In *The Twelfth International Conference on Learning Representations*.

George A. Miller. 1995. Wordnet: a lexical database for english. *Commun. ACM*, 38(11):39–41.

Seyed Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2025. GSM-symbolic: Understanding the limitations of mathematical reasoning in large language models. In *The Thirteenth International Conference on Learning Representations*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024. MMLU-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Peter Cathcart Wason. 1960. On the failure to eliminate hypotheses in a conceptual task. *Quarterly Journal of Experimental Psychology*, 12:129 – 140.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. 2025. Browsecomp: A simple yet challenging benchmark for browsing agents. *Preprint*, arXiv:2504.12516.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

A When Can a Hypothesis be Falsified?

We provide a detailed analysis of when a test yields a falsification depending on the relation between the current hypothesis H and the target rule R. Following Klayman and Ha (1987), we classify any test x along two binary dimensions: whether $x \in H$ (the test is *positive* with respect to the hypothesis) and whether $x \in R$ (the test is *true* with respect to the target). A test within H is a *positive* test; a test outside H is a *negative* test. Whether the oracle confirms or rejects it determines whether the test is *true* or *false*. This yields four cases, summarized in Table 2, each with a distinct informational value for hypothesis revision.

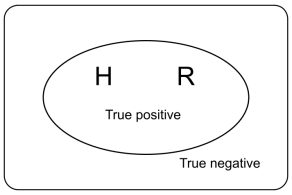
	$x \in R$	$x \notin R$
$x \in H$	True positive	False positive
$x \notin H$	False negative	True negative

Table 2: Classification of a test x by membership in H and R.

In the following sections, we discuss in each possible relation between H and R which of the four quadrants yield a conclusive falsification and which do not.

Case 1: $H = R$ (Identical)

When H is identical to R, both false positives and false negatives are impossible. Every member of H is in R (so no test within H can receive "Do not conform"), and no member outside H is in R (so no test outside H can receive "Conform"). Trivially, holding the correct hypothesis means that it cannot be falsified and only a `Guess` action would end the game.

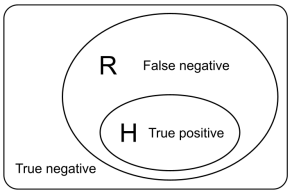


Case 2: $H \subset R$ (H too narrow)

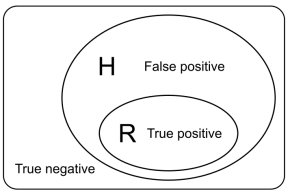
When H is a strict subset of R, every member of H is automatically in R, making false positives impossible: tests within H always receive "Conform". The only route to falsification is a false negative – testing outside H and receiving "Conform", revealing that R extends beyond H and forcing upward revision. Crucially, not every negative test achieves this: testing outside H and receiving "Do not conform" (true negative) is equally consistent with H and provides no new information. A player relying exclusively on positive tests is therefore confined to true positives and can never falsify H. This is the dominant relation in FALSIFYBENCH.

Case 3: $H \supset R$ (H too broad)

When H is strictly broader than R, every member of R is inside H, making false negatives impossible: no test outside H can receive "Conform". The only route to conclusive falsification is a false

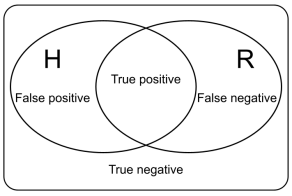


positive – testing within H and receiving "Do not conform", revealing that H must be narrowed. Notably, a positive test strategy is the correct approach in this configuration: testing within H is the only way to a falsification. Crucially, not every positive test achieves this: testing within H and receiving "Conform" (true positive) is equally consistent with H and provides no reason to discard it.



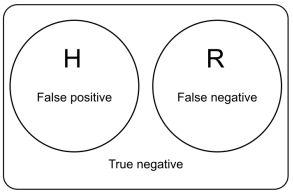
Case 4: H and R partially overlap

When H and R partially overlap but neither contains the other, both false positives and false negatives are accessible. A test within H but outside R yields "Do not conform" (false positive, falsification): only a subset of H is compatible with R, revealing that H must be revised to exclude that region. A test within R but outside H yields "Conform" (false negative, falsification): R extends beyond H, revealing that H must be revised to include that region. Thus, two ways of falsification are available in this configuration. However, two non-conclusive outcomes are also possible: testing within both H and R yields "Conform" (true positive, non-conclusive), and testing outside both H and R yields "Do not conform" (true negative, non-conclusive). In both cases the feedback is consistent with H and provides no pressure to revise it in either direction.



Case 5: H and R disjoint

When H and R share no members, every test within H yields "Do not conform" (false positive, falsification): H is entirely incompatible with R and must be discarded. Every test within R but outside H yields "Conform" (false negative, falsification): R extends into a region entirely outside H, and H needs to be revised. The only non-conclusive outcome is testing outside both H and R, which yields "Do not conform" (true negative, non-conclusive): the feedback is consistent with H and provides no information about how to revise it. Finally, true positives are impossible in this configuration, since H and R share no members.



B Game Construction Details

We build game instances using the WordNet taxonomy, where each node is a semantic category and its descendants represent increasingly specific subcategories. As we explained in Section 3.1, we started with seven high-level WordNet categories representing concrete concepts as candidate target rules: "animal", "artifact", "body part", "food", "location", "plant", and "worker". However, many of the possible descendants of these categories involve low-frequency vocabulary. Hence, we manually filtered the subcategory pool, selecting only five target categories: "animal", "artifact", "body part", "food" and "plant".

Since subcategories of these target categories can come from different depths in the taxonomy (e.g. "feline" is deeper than "mammal"), we first divided the subcategories into two groups based on their taxonomic distance from the targets. The *close* group has a distance of at most four edges, and the *deep* group exceeds four edges. To ensure diversity in the sampling categories for each game, we sampled triples from the close and deep categories in equal proportions. Table 3 lists all the selected sampling categories for the close and deep categories, respectively, grouped by target rule.

C Game Engine Details

C.1 Prompts

The prompts we used verbatim in all experiments are shown in Tables 5–7. In each prompt, double curly-brace placeholders (e.g. `{{max_turns}}`) are

Target rule	Close categories (≤ 4)	Deep categories (> 4)
animal	aquatic vertebrate, bird, mammal, vertebrate	Australian magpie, New World blackbird, New World flycatcher, New World oriole, New World sparrow, New World vulture, New World warbler, Old World vulture, Old World warbler, bear, big cat, bird of prey, bony fish, cat, dog, eagle, falcon, feline, fox, goose, hawk, lion, owl, shark, sparrow, sunfish, swallow, tiger, vulture, whale, wolf
artifact	bag, clothing, container, electrical device, electronic device, musical instrument, utensil	aircraft, airplane, bioweapon, blanket, chamber, church, cooling system, cutter, cutting implement, engine, guitar, hammer, horn, lamp, measuring instrument, missile, motor, painting, pedal, pottery, propeller, robe, ship, stringed instrument, tube, valve, vessel, warship, weapon, wind instrument
body part	internal organ, muscle, reproductive organ, sense organ	artery, blood vessel, bone, conjunctiva, facial muscle, mucous membrane, nose, ocular muscle, skeletal muscle, vein, venous sinus, vesicle
food	appetizer, bread, cake, dessert, meal, pastry, pie, soup	bacon, beefsteak, biscuit, broth, candy, chicken, cocktail, condiment, conserve, cookout, cornbread, dip, fillet, flatbread, frozen dessert, fudge, goulash, ice cream, muffin, pudding, sugar, sweet roll, tart, toast
plant	angiosperm, aquatic plant, gymnosperm, herb, tree, woody plant	anemone, anise tree, arborvitae, barberry, cedar, clematis, columbine, conifer, custard apple, cycad, cypress, fir, fruit tree, hemlock, larch, laurel, magnolia, pine, silver fir, spruce, white pine, wildflower

Table 3: Selected sampling categories grouped by target rule and taxonomic distance from the target. Close categories have distance ≤ 4 in the WordNet taxonomy, while deep categories have distance > 4.

filled before each game. The player prompts are shown in Table 5; the oracle prompts are shown in Table 6. Additionally, Table 7 shows the prompt used for the turn level annotation of the relations between a current hypothesis and the target rule.

C.2 Agent Roles and Action Space

Each game began with exactly three positive examples presented to the player. The player was never shown the target rule R or the sampling category S directly; it observed only the three example items and the system prompt explaining the rules of the game (see Table 5).

The interaction proceeded as a sequence of turns between the player and the oracle. The player was responsible for proposing actions aimed at discovering the hidden property, while the oracle evaluated these actions and provided feedback according to the player's last action.

On each turn, the player performed exactly one of two actions:

- **Test:** propose a triple of items together with a hypothesis string and a reasoning trace. The action must be well-formed JSON matching the schema specified in the system prompt; malformed or schema-invalid submissions trigger a retry (see Appendix C.3) and are not recorded as turns.
- **Guess:** submit a single property string as a final answer. A correct guess returns "Correct" and ends the game; an incorrect guess returns "Incorrect", after which the player can continue with additional `Test` or `Guess` actions.

After receiving a valid player action, the oracle performed the following actions:

- **Feedback generation**: for a `Test` action, determine whether the proposed items satisfy the target rule R and return the string "Conform" if they do and "Do not conform" otherwise; for a `Guess` action, determine whether the proposed property is semantically equivalent to R by returning the string "Correct" if it does and "Incorrect" otherwise.
- **Strategy classification**: after every `Test` action, classify the player's testing strategy as *positive* if the player tests examples expected to satisfy its current hypothesis, and *negative* if it tests examples outside the hypothesis in order to challenge it and explore broader alternatives. This step was performed only for later analysis and did not affect gameplay directly.

A game turn was defined as one accepted player action together with the oracle's response. Invalid submissions and retry attempts did not increment the turn counter.

C.3 Retry Mechanism

The game engine employed two independent retry mechanisms to handle failures in producing valid JSON outputs, and none of these retry attempts were recorded as game turns. For player actions, if the engine received malformed JSON or an action that violated the schema, it re-prompted within the same turn requesting strict JSON output; after the first failure, up to three additional retries were permitted. For oracle judgments, each method was allowed up to two attempts: if the first response could not be parsed as JSON, the engine re-issued the prompt with an appended instruction ("IMPORTANT: Return ONLY a JSON object") and then attempted to parse the second response. This numeber of retries was sufficient to complete all games with all tested models.

C.4 Memory and statefulness.

During the game interactions, the player was *stateful*: its conversation history accumulated all prior oracle feedback and its own previous actions. In contrast, the oracle was *stateless*: each call to the oracle was performed using a newly initialized prompt that provided only the elements necessary to collect the current judgement, with no access to information from earlier turns.

D Experiment Details

D.1 LLMs Details

The following are the model identifiers of all evaluated LLMs:

Identifier
gpt-5.2-chat
gpt-5-mini
gpt-5-nano
openai/gpt-oss-120b
openai/gpt-oss-20b
deepseek-ai/DeepSeek-V3.1
Qwen/Qwen3.5-9B
MiniMaxAI/MiniMax-M2.5
meta-llama/Llama-4-Maverick-17B-128E-Instruct-FP8
mistralai/Mistral-Small-24B-Instruct-2501
moonshotai/Kimi-K2.5
zai-org/GLM-5

Among these models, `gpt-5.2-chat` was used via the Azure OpenAI API, while `gpt-5-mini` and `gpt-5-nano` were used via the standard OpenAI API. All other models were evaluated using the TogetherAI API.

When used in the oracle role, reasoning models were run with either low reasoning or no reasoning, depending on the model. When used in the player role, reasoning models were run at a medium level of effort. When the player failed to produce a valid JSON action, low reasoning effort was requested to speed up generation during retries. Regarding temperature and other sampling parameters, we used the provider defaults for all instruction-tuned models. For reasoning models, parameters were adapted according to provider-specific instructions depending on whether the model was used in reasoning or non-reasoning mode. The total cost of running all experiments across all models amounted to approximately $150.

D.2 Mixed-Effects Logistic Regression Details

The experimental design of FALSIFYBENCH assigned each LLM to both the player and oracle roles. As reported in Section 4, we performed a game-level analysis to estimate how player strategy and oracle feedback relate to player success. Since we had 12 LLMs and each played 100 games, we obtained a total of 1,200 game-level observations. We fit a mixed-effects logistic regression using variational Bayes inference of the form:

$$\begin{aligned}\text{logit}\, P(\text{success}_{ij}) = \alpha \\ + \beta_1 \cdot \text{confirmation_bias}_{ij} \\ + \beta_2 \cdot \text{oracle_error}_{ij} \\ + u_j\end{aligned} \tag{1}$$

where confirmation bias is a continuous variable in the interval $[0, 1]$, oracle error is a binary variable in $\{0, 1\}$, i indexes games, j indexes models, and $u_j \sim \mathcal{N}(0, \sigma_u^2)$ is an LLM-level random intercept. The model was fit using variational Bayes inference with `BinomialBayesMixedGLM` from the `statsmodels` Python package. We report posterior means, posterior standard deviations, and implied odds ratios for all fixed effects as follows:

Fixed Effects
`Intercept: 1.955 (SD 0.066), OR 7.067`
`Confirmation bias: -4.148 (SD 0.116), OR 0.016`
`Oracle error: -0.123 (SD 0.147), OR 0.885`
Random Effects
`LLM SD` ($\hat{\sigma}_u$)`: 0.553`

Rule	Trigger keywords / pattern
SYLLABLE	*syllable*, *syllables*, *syllabic*
LETTER	*letter*, *letters*
WORD_COUNT	*single-word*, *two word*, *multiple words*, ...
PREFIX_SUFFIX	*prefix*, *suffix*, *prefixes*, *suffixes*
VOWEL_CONSONANT	*vowel(s)*, *consonant(s)*
SPELLING	*spelling*, *spelled*, *spelt*, *orthography*, *orthographic*
PRONUNCIATION	*pronounce*, *pronounced*, *pronunciation*, *phonetic*, *rhyme*, *rhymes*, *rhyming*
NAMING	*common name*, *scientific name*, *latin name*, *names have*
ALPHABETIC	*alphabet*, *alphabetic*, *alphabetical*
INITIAL_FINAL_UNIT	*first letter*, *last word*, *initial syllable*, *final word*, ...
META_LING. STRUCT.	{*starts*, *begins*, *ends*, ...} × {*letter*, *word*, *name*, *syllable*, ...}

Table 4: Patterns used in the surface-level linguistic feature classifier. Each rule is triggered when the hypothesis text matches the listed keywords or pattern. A single hypothesis may match multiple rules simultaneously.

In this analysis the posterior standard deviation is used as a measure of estimation uncertainty. An approximate 95% credible interval is obtained using a normal approximation $\mu \pm 1.96\sigma$.

Confirmation bias was a strong negative predictor of success. Since Confirmation bias ranges from 0 (no Confirmation bias) to 1 (maximum Confirmation bias), the corresponding odds ratio of 0.016 reflects the effect of moving across the full range of the variable, reducing the odds of success to approximately $1/63$ of their original value. The 95% credible interval for the log-odds coefficient is $[-4.148 \pm 1.96 \times 0.116] = [-4.375, -3.921]$, entirely below zero, confirming a robust negative effect.

By contrast, oracle error had a negligible and highly uncertain effect. The implied odds ratio of 0.885 corresponds to an approximate 11.5% reduction in the odds of success when an oracle error is present, though this effect is small and the credible interval $[-0.123 \pm 1.96 \times 0.147] = [-0.411, 0.165]$ spans zero, indicating that the effect is not credibly different from zero.

The random-effect standard deviation of $\hat{\sigma}_u = 0.553$ indicates moderate residual between-model variability not explained by the two predictors. This is expected, as model identity captures additional factors that generally determine underlying LLM quality, such as model size, amount of training data, and post-training procedures, none of which are explicitly modeled in the fixed effects.

D.3 Linguistic Feature Classifier Details

To detect hypotheses that focus on surface-level linguistic properties of words rather than their semantic content, we designed a heuristic classifier based on regular expressions. The classifier comprises eleven rules, each targeting partially distinct types of surface-level property (see Table 4). A hypothesis was classified as linguistic if it matched *at least one* rule. Multiple rules could be triggered on the same hypothesis: for example, the hypothesis "The items all have two-word common names and start with the same letter" simultaneously matches LETTER, WORD_COUNT, NAMING, and META_LINGUISTIC_STRUCTURE. Consequently, per-rule counts reflect the number of linguistic-pattern matches across all hypotheses (with overlaps permitted), not the number of distinct games or turns. Figure 7 shows the per-rule counts across all game turns.

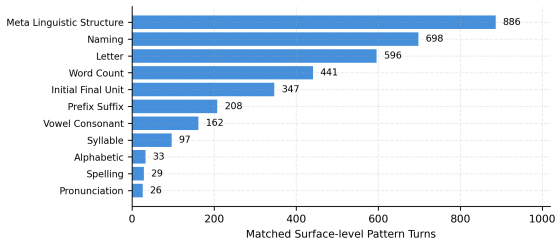


Figure 7: Raw match count between surface-level linguistic patterns on game turns.

E Additional Results

E.1 Additional Player Metrics

In addition to the player metrics presented in the main paper, we also show here two additional metrics that are informative for understanding the overall quality of players.

Average Turns to Solution. The average number of turns required to correctly identify the target rule R, computed over successful games only.

Average Guesses per Game. The average number of `Guess` actions submitted per game. While

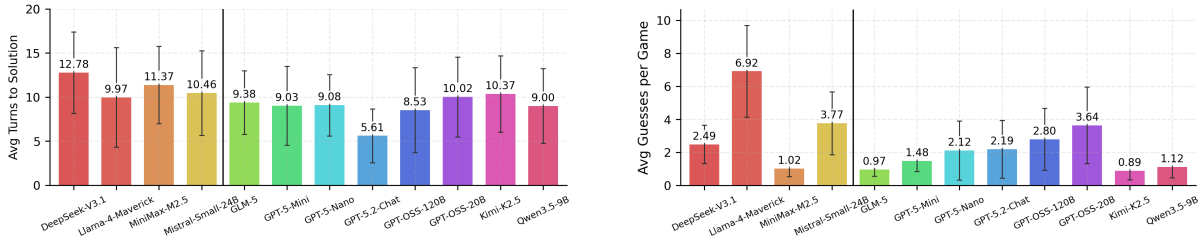


Figure 8: **Left:** The average number of turns required to correctly identify the target rule R, computed over successful games only. **Right:** The average number of `Guess` actions performed per game. In both plots, models to the left of the vertical black line are **instruction-tuned** variants; models to the right are **reasoning** variants.

guessing can reveal that a hypothesis is wrong by receiving the string "Incorrect" as feedback, it reflects a passive strategy compared to targeted `Test` actions designed to probe the boundaries of the current hypothesis.

Figure 8 reports the results for each model on these two additional metrics. In terms of turn efficiency, most models fell in the range between 8 and 11 turns, and we did not observe as large a gap between reasoning and instruction-tuned models as we did for other metrics, such as success rates. A notable positive exception is GPT-5.2-Chat, which stood out as the most efficient model overall, solving games in just 5.61 turns on average. Regarding the average guesses per game, Llama-4-Maverick stood out as the model with the most passive strategy, combining one of the highest positive testing bias scores (81.8%, see Figure 2) with the highest average number of guesses per game (6.92), suggesting it relied on repeated guessing rather than active falsification to make progress. At the other extreme, the two largest open models, GLM-5 and Kimi-K2.5, showed the lowest guessing rates (GLM-5: 0.97, Kimi-K2.5: 0.89), indicating that they did not commit to hypotheses prematurely.

E.2 Relation Distribution in Successful and Failed Games

Figures 9 shows the distribution of relations between H and R across `Test` turns, split by whether the game was ultimately successful or not. In both conditions, $H \subset R$ (Hypothesis in Target) was the dominant configuration, confirming that models generally maintained hypotheses that were too narrow throughout the game. However, noticeable differences emerged between the two groups.

In successful games, the distribution was more concentrated around $H \subset R$, suggesting that successful play tended to follow a clean upward revision trajectory that mirrored the structure of the hierarchical hypothesis space derived from the WordNet taxonomy.

In failed games, the partial overlap configuration was the second most common relation for most models, particularly among stronger models: GPT-5.2-Chat reached nearly 40% Partial Overlap turns, while GPT-OSS-120B and GLM-5 reached approximately 30% and 25%, respectively. This suggests that, in failed games, these models frequently arrived at hypotheses similar to R but failed to follow a clear upward path. The proportion of disjoint cases also tended to be higher in failed games, though to a lesser extent, suggesting that fully off-track hypotheses were associated with failure. Finally, stronger models also exhibited cases in which they had effectively reached the target rule but failed to commit to a `Guess`, either abandoning it prematurely or exhausting the turn limit.

E.3 Decomposition by target rule

Figures 10 and 11 report success rate, Confirmation bias, conclusive falsification rate, average turns to solution, and average guesses per game decomposed by target rule ($R \in$ {"animal", "artifact", "body part", "food", "plant"}). The most consistent finding is that "artifact" was by far the hardest target rule across all models and metrics: success rates were uniformly low (0–30% for most models), Confirmation bias was highest, and conclusive falsification rates were lowest. This was likely because artifact subcategories spanned a semantically heterogeneous space, making it harder to form hypotheses that generalized cleanly upward through the taxonomy. By contrast, "animal" was the easiest target for most reasoning models, with GPT-5.2-Chat achieving a 100% success rate and several others exceeding 75%.

Regarding average guesses per game, Llama-4-Maverick submitted an average of nearly 8 guesses across all target properties, far exceeding every

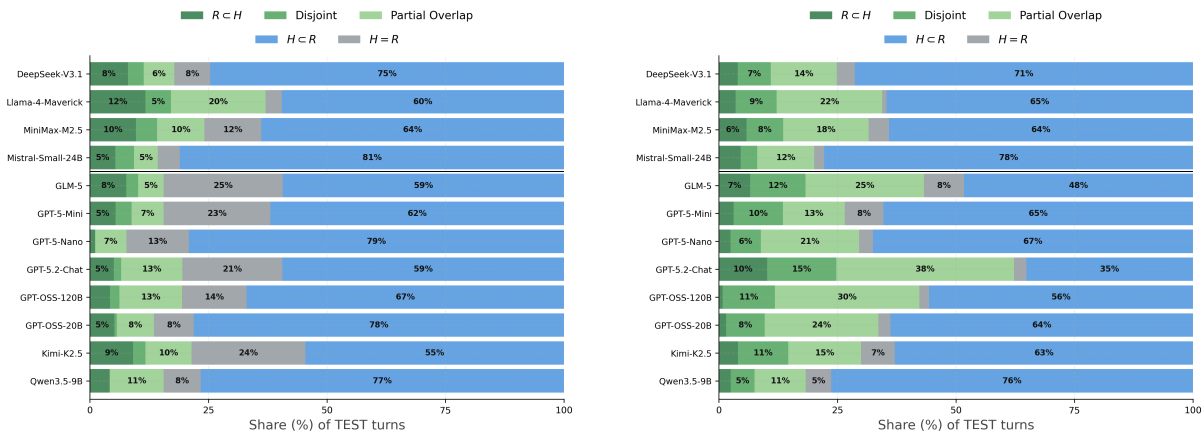


Figure 9: Distribution of relations between H and R across `Test` turns, split by outcome. **Left:** Successful games only. **Right:** Failed games only. Successful games are dominated by the $H \subset R$ configuration (blue) reflecting coherent upward revision. Failed games show a substantially higher proportion of Partial Overlap (green) and Disjoint (yellow) turns across most models, indicating that failure is most commonly associated with failure in exploring the hierarchical structure of the hypothesis space.

other model and confirming that its strategy relied heavily on passive guessing rather than active evidence gathering regardless of the semantic domain. In the artifact category, well-performing models also showed elevated guessing behavior, suggesting that the semantic difficulty of this category induced more premature commitment across models. Turn efficiency broadly mirrored success rate patterns, with *animal* games being solved fastest by reasoning models and *artifact* games either taking the most turns or remaining unsolved within the turn limit, as indicated by empty cells in the turns heatmap for models with zero success in that category.

E.4 Decomposition by Outcome

Figure 13 reports the average guesses per game and Confirmation bias split by game outcome (Succeeded vs. Failed).

In terms of average guesses (left panel), failed games involved substantially more guesses than successful ones across all models. The effect was most pronounced for Llama-4-Maverick, which averaged 8.5 guesses per failed game compared to 3.8 in successful ones, and for GPT-5.2-Chat, GPT-OSS-120B, and GPT-OSS-20B, which all approached 4–5 guesses per failed game while remaining below 2 in successful ones. This suggests that models relied on guessing and "Incorrect" feedback to disconfirm hypotheses when they were not able to efficiently explore the hypothesis space.

Similarly, regarding Confirmation bias (right panel), failed games exhibited higher Confirmation bias than successful ones for nearly every model. The gap was largest for stronger reasoning models: GPT-5.2-Chat showed 20% bias in successful games versus 50% in failed ones, while GPT-OSS-20B showed 17% versus 53%. This provides further within-model evidence that negative testing was associated with success and represented the most efficient strategy for reaching the target.

E.5 Qualitative Game Examples

To give the reader a better understanding of the game dynamics, we present one successful (Table 8) and one failed (Table 9) complete game trace produced by GPT-5.2-Chat. In each trace the player's strategy in `Test` turns is labeled with *positive* for positive tests and *negative* for negative tests.

F Use of Generative AI Tools

We used GitHub Copilot and Claude Code as assistant tools for parts of the project's source code development, while ChatGPT was used to correct minor grammatical errors in the manuscript.

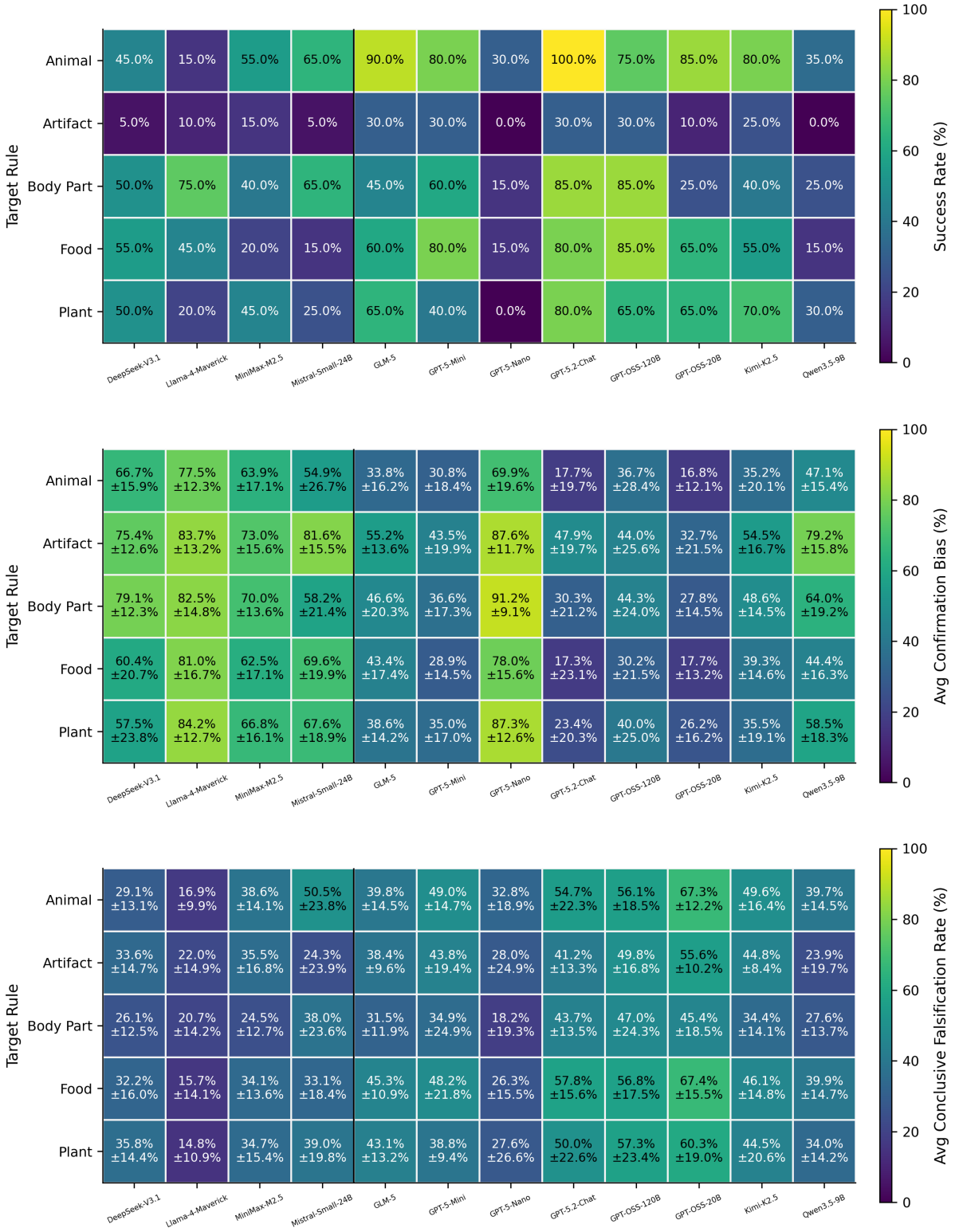


Figure 10: **Top:** success rate. **Center:** average Confirmation bias. **Bottom:** average conclusive falsification rate. All metrics are decomposed by target rule.

Figure 11: **Top:** average turns to solution. **Bottom:** average guesses per game. All metrics are decomposed by target rule. Empty cells in the turns heatmap indicate zero successful games for that model-target rule combination.

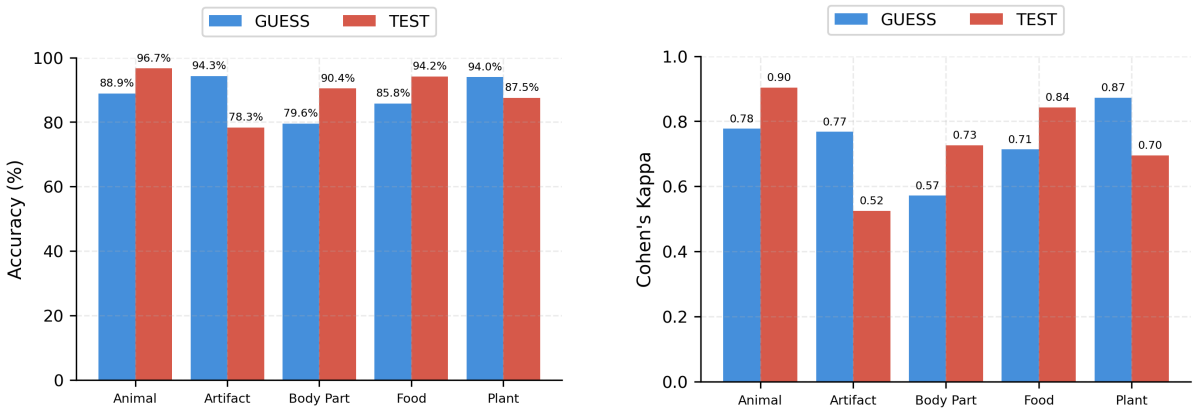


Figure 12: Oracle accuracy and agreement with human annotator decomposed by target rule. **Left:** Raw accuracy. **Right:** Cohen's κ agreement.

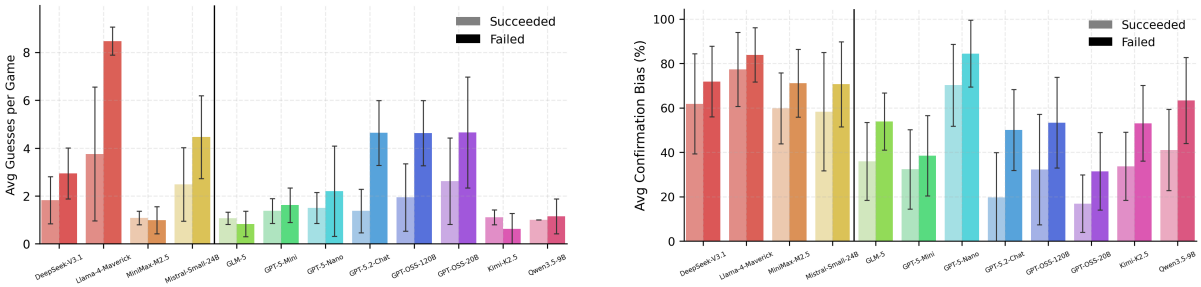


Figure 13: Results decomposed by game outcome (Succeeded vs. Failed). **Left:** Average guesses per game. **Right:** Average positive testing bias. Light bars correspond to successful games; dark bars to failed games. Error bars show standard deviation.

Player System Prompt

```
You are playing a property discovery game. You are given 3 items that share a common property.
Your goal is to guess the property by testing new items.

You have a maximum of {{max_turns}} turns.

What to expect from the oracle:
- After a Test action, the oracle will reply with: "Oracle's judgment: CONFORM" or "Oracle's
judgment: DO NOT CONFORM".
- After a Guess action, the oracle will reply with: "Oracle's judgment: INCORRECT" if you're
wrong; if you're correct the game ends.

In each turn, you can perform one of two actions:
  1. Propose 3 new items to test your current hypothesis.
  2. Make a guess of the property if you think you know it.

The purpose of testing is to gather information to refine your hypothesis. Once you believe
you have enough information, you can make a guess. If your guess is correct, you win. If it's
incorrect, you can keep testing and guessing until you find the right property or run out of
turns.

Your response must be valid JSON in one of these formats:

For testing:
{
    "action": "test",
    "items": ["item1", "item2", "item3"],
    "hypothesis": "your current hypothesis",
    "reasoning": "explain your thinking"
}

For guessing:

{
    "action": "guess",
    "property": "your final guess as a single word or phrase",
    "reasoning": "explain why you're confident"
}
```

Player's first user message

```
Initial examples that share a common property:
  1. {{initial_examples[0]}}
  2. {{initial_examples[1]}}
  3. {{initial_examples[2]}}

What is your first move? Respond with JSON only.
```

Table 5: System prompt and first user role interaction provided to the player. The system prompt specifies the rules of the game and is provided once at the start of each game; the initial message introduces the three initial examples. Double curly-brace placeholders are filled before each game.

Judge `Test` triples against target rule

```
You are an oracle in a property discovery game.
The target rule is: “{{target_property}}”
The player has proposed these items:
  1. {{items[0]}}
  2. {{items[1]}}
  3. {{items[2]}}
Do ALL three items conform to the property “{{target_property}}”?
Respond with ONLY a JSON object:
{
    "conforms": true or false,
    "explanation": "brief explanation of your judgment"
}
```

Classify player’s test strategy

```
You are an oracle in a property discovery game.
The property to evaluate is: “{{property_hypothesis}}”
The player has proposed these items:
  1. {{items[0]}}
  2. {{items[1]}}
  3. {{items[2]}}
Player’s current hypothesis:
“{{property_hypothesis}}”
Player’s reasoning for choosing these items:
“{{player_reasoning}}”
Tasks:
1) Infer the player’s *intention* for this Test based on their hypothesis and reasoning.
    - "confirm" = they expect all three items to conform to their hypothesis
    - "falsify" = they expect at least one item to NOT conform to their hypothesis
2) Separately, judge whether all three items actually conform to the hypothesis.
Respond with ONLY a JSON object:
{
    "intention": "confirm" or "falsify",
    "expected_conforms": true or false,
    "conforms": true or false,
    "explanation": "brief explanation"
}
```

Judge `Guess` against the target rule

```
You are an oracle in a property discovery game.
The target rule is: “{{target_property}}”
The player has guessed: “{{guessed_property}}”
Are these semantically equivalent (same meaning, possibly different wording)?
Respond with ONLY a JSON object:
{
    "correct": true or false,
    "explanation": "brief explanation"
}
```

Table 6: Complete prompts provided to the oracle. Each prompt specifies a different type of annotation that the oracle has to provide. Double curly-brace placeholders are filled before each game.

Annotation Prompt

```
You are annotating saved game turns from an inductive reasoning experiment.

Treat the target rule and the current hypothesis as semantic sets or classes of entities. Decide
the relation between those two sets using exactly one label from this closed set:

- identical: the two sets are the same.
- disjoint: the two sets share no members.
- partial_overlap: the two sets overlap, but neither includes the other.
- hypothesis_included_in_target: the hypothesis set is a proper subset of the target set.
- target_included_in_hypothesis: the target set is a proper subset of the hypothesis set.

Interpret terms by their normal semantic meaning, not by string overlap. If one label is more
specific than another, choose the more specific label:
- Prefer identical over either inclusion label when the sets are the same.
- Prefer an inclusion label over partial_overlap when one set fully contains the other.

Examples using target rule "animal":
1. Target: "animal" | Hypothesis: "animal" -> identical
2. Target: "animal" | Hypothesis: "mammal" -> hypothesis_included_in_target
3. Target: "animal" | Hypothesis: "living thing" -> target_included_in_hypothesis
4. Target: "animal" | Hypothesis: "carnivore" -> partial_overlap
5. Target: "animal" | Hypothesis: "mineral" -> disjoint

Now annotate this case.
target rule: "{{target_property}}"
Current hypothesis: "{{hypothesis}}"

Respond with JSON only in this format:

{
    "relation": "one of identical, disjoint, partial_overlap, hypothesis_included_in_target,
target_included_in_hypothesis",
    "explanation": "at most 12 words"
}

Do not add any extra keys, markdown, or prose outside the JSON object.
```

Table 7: System prompt used for semantic set-relation annotation. The annotator classifies the relationship between a target rule and a hypothesis using a fixed label set. An explanation is included to better understand the provided annotation. Double curly-brace placeholders are filled before each API call.

Successful Game

target rule (hidden to player): body part
Sampling Category (hidden to player): sense organ
Initial Items: thermoreceptor, lateral line, third eye

T1 *(positive)* **Items:** electroreceptor, Jacobson's organ, ampullae of Lorenzini **H:** Specialized sensory organs/receptors that detect environmental stimuli. **R:** Similar to the initial examples; testing other exotic sensory structures. → `CONFORM`

T2 *(negative)* **Items:** retina, cochlea, neuron **H:** Specialized anatomical structures dedicated to detecting specific sensory stimuli. **R:** Including a neuron—a general nerve cell, not a sense organ—to probe the property's boundary. → `DO NOT CONFORM`

T3 *(negative)* **Items:** magnetoreceptor, infrared pit organ, taste bud **H:** Specialized sensory organs providing non-classical senses beyond the standard five. **R:** Testing whether organs tied to well-known senses (taste bud) also conform. → `CONFORM`

T4 *(guess)* **Property:** "sensory organs or sensory receptors" **R:** All tested sensory structures conform; neuron does not. → `INCORRECT`

T5 *(negative)* **Items:** retina, cochlea, olfactory epithelium **H:** Sensory structures that are not the primary classical human sense organs. **R:** Testing canonical sensory organs to see if they conform despite belonging to the five standard senses. → `CONFORM`

T6 *(negative)* **Items:** heart, kidney, lung **H:** Anatomical structures specifically involved in sensory perception. **R:** Testing organs with purely physiological (non-sensory) roles to check whether the property is broader than sensory function. → `CONFORM`

T7 *(guess)* **Property:** "anatomical body parts" **R:** Non-sensory organs (heart, kidney, lung) still conform; the property must cover all body parts. → `CORRECT`

SUCCESS ✓

Table 8: Complete game trace for a successful game. The oracle feedback follows the → sign. The model correctly identifies the target in 7 turns. **T** = turn; **H** = current hypothesis; **R** = reasoning; *positive* = positive test; *negative* = negative test; *guess* = guess turn.

Failed Game

target rule (hidden to player): artifact
Sampling Category (hidden to player): electrical device
Initial Items: relay, primary coil, solenoid

T1 *(positive)* **Items:** transformer, inductor, electromagnet **H:** Devices that operate using electromagnetic coils. **R:** All initial examples are coil-based devices; testing others of the same kind. → `CONFORM`

T2 *(negative)* **Items:** capacitor, resistor, diode **H:** Electrical components relying on electromagnetic coils or inductive windings. **R:** Testing components that do not use coils to check whether coil-dependency is required. → `CONFORM`

T3 *(negative)* **Items:** gear, spring, pulley **H:** Electrical or electronic components. **R:** Testing purely mechanical components to probe whether the property is restricted to electrical items. → `CONFORM`

T4 *(negative)* **Items:** happiness, freedom, velocity **H:** Tangible physical objects (not abstract concepts). **R:** Testing abstract concepts to verify whether tangibility is required. → `DO NOT CONFORM`

T5 *(guess)* **Property:** "tangible physical objects" **R:** Abstract concepts do not conform; all physical objects tested so far do. → `INCORRECT`

T6 *(positive)* **Items:** apple, mountain, table **H:** The word has exactly two syllables. **R:** Testing whether syllable count explains the conformance pattern. → `DO NOT CONFORM`

T7 *(positive)* **Items:** engine, bolt, circuit **H:** Technical or engineering-related terms. **R:** All initial examples are engineering terms; testing whether this domain restriction defines the property. → `CONFORM`

T8 *(guess)* **Property:** "engineering-related terms" **R:** All conforming items are engineering or technical terms. → `INCORRECT`

T9 *(positive)* **Items:** keyboard, wheel, processor **H:** Components or parts of a larger machine or system. **R:** All examples so far are parts of larger systems; testing other machine components. → `CONFORM`

T10 *(negative)* **Items:** rock, tree, river **H:** Man-made components or parts of engineered systems. **R:** Testing natural, non-manufactured objects to check whether being man-made is required. → `DO NOT CONFORM`

T11 *(guess)* **Property:** "man-made components of engineered systems" **R:** Natural objects do not conform; all man-made items tested so far do. → `INCORRECT`

T12 *(negative)* **Items:** chair, car, computer **H:** The word contains the letter 'o'. **R:** Testing a letter-based pattern noticed across many conforming words. → `CONFORM`

T13 *(guess)* **Property:** "contains the letter 'o' " **R:** Chair, car, computer all contain 'o' and conform. → `INCORRECT`

T14 *(negative)* **Items:** desk, sofa, lamp **H:** A machine or mechanical/electrical device (including components). **R:** Testing man-made everyday objects that are not machines, to probe whether device-hood is required. → `CONFORM`

T15 *(negative)* **Items:** cat, dog, bird **H:** Inanimate (non-living) things. **R:** Testing living organisms to determine whether animacy is the distinguishing factor. → `DO NOT CONFORM`

T16 *(guess)* **Property:** "non-living things" **R:** Living organisms do not conform; all non-living items tested so far do. → `INCORRECT`

T17 *(negative)* **Items:** sky, ocean, stone **H:** The word has exactly one syllable. **R:** Re-examining whether syllable count distinguishes conforming from non-conforming items. → `DO NOT CONFORM`

T18 *(negative)* **Items:** bus, train, plane **H:** The word ends with a consonant. **R:** Testing a phonological pattern as a potential systematic rule. → `CONFORM`

T19 *(guess)* **Property:** "countable nouns" **R:** All conforming items are countable nouns; non-conforming items include abstractions and natural phenomena. → `INCORRECT`

T20 *(positive)* **Items:** oxygen, gold, water **H:** The word contains the letter 'e'. **R:** A final letter-based attempt, noting that many conforming words contain 'e'. → `DO NOT CONFORM`

Turn limit reached – FAILED ×

Table 9: Complete game trace for a failed game. The oracle feedback follows the → sign. The game ends after the model used the 20-turn limit without identifying the target. Abbreviations as in Table 8.